Available online at www.sciencedirect.com
ScienceDirect
SoftwareX 5 (2016) 89–95
SoftwareX
www.elsevier.com/locate/softx

Ocelet: Simulating processes of landscape changes using interaction graphs

P. Degenne, D. Lo Seen*

Cirad, Environment and Societies Department, UMR TETIS, Montpellier, France

Received 9 February 2016; received in revised form 6 April 2016; accepted 10 May 2016

Abstract

This paper introduces Ocelet, a domain specific language and simulation tool for modelling changes in geographical landscapes. It is characterised by the use of *interaction graphs* (graphs with interaction functions on their edges) to represent the system as composed of processes, each involving several entities distributed in space that are in interaction with each other. Entities are the vertices of the graphs, and interactions are the edges on which (interaction) functions can be applied to make the system change through time. Examples are given to illustrate the generic disposition of the simulation approach to model and study changing geographical setups.

Keywords: Spatial modelling; Multi-scale; Domain specific language

Code metadata

Current code version	*v1.1*
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-16-00024
Legal Code License	*CeCILL v2.1(GNU GPL compatible Free Software licence agreement designed byFrench Public Research Institutes)* Licence detailed information: http://www.cecill.info/licences/Licence_CeCILL_V2.1-en.html
Code versioning system used	*Git*
Software code languages, tools, and services used	*Java 7, Eclipse, Xtext*
Compilation requirements, operating environments & dependencies	*Development and compilation : Eclipse Luna, Java 7 Dependencies : Eclipse RCP, Xtext-XBase 2.8, Geotools 9.4, JAK*
If available Link to developer documentation/manual	*NA*
Support email for questions	*ocltdev@ocelet.fr*

Software metadata

Current software version	*Carmine (1.1)*
Permanent link to executables of this version	*Download tab on* http://www.ocelet.org/
Legal Software License	*CeCILL V2.1*
Computing platforms/Operating Systems	*Linux, OS X, Windows*
Installation requirements & dependencies	*Java 7 (or 8) Runtime Environment*
If available, link to user manual - if formally published include a reference to the publication in the reference list	Documentation tab on http://www.ocelet.org/
Support email for questions	*ocltdev@ocelet.fr*

1. Motivation and significance

Simulation models are used both as research tools, to test hypotheses when trying to improve the understanding of a

* Corresponding author.
E-mail addresses: loseen@teledetection.fr, danny.lo_seen@cirad.fr (D. Lo Seen).

http://dx.doi.org/10.1016/j.softx.2016.05.002

system, and in decision support, to explore alternative scenarios (e.g. [1]). But modelling the environment as a system can be considered particularly challenging as several interacting processes often need to be modelled. These processes can also be dynamic, spatially distributed, at several scales of space and time, and may involve human activities [2]. Several reviews of available methods for modelling the environment exist (e.g. [3–5]). Most of the methods belong to three main approaches that stand out by the size of their user communities: Systems Dynamics (SD), Cellular Automata (CA) and Agent-based modelling (ABM). The SD approach proposed by Forrester [6,7] represents real-world processes in terms of stocks (system variables), flows (exchanges between stocks) and interacting feedback loops (an output of the system can be fed back as input to the system). Examples of software based on these principles include STELLA [8] and Vensim [9]. But when a system is distributed in a geographical space, aggregated system variables become inadequate. The solution proposed by the Spatial Modelling Environment (SME—[26]) was to disaggregate the system space into cells. Stock-flow models could then be included in each of the cells, with neighbouring cells able to exchange flows. Referred to as "individual-based modelling approaches", CA and ABM are inherently different in that aggregate patterns emerge from the sum of individual behaviour [10]. With CA, geographical space is represented by grid cells that can take a finite number of states. The state of a given cell changes following transition rules that depend on the states of the neighbouring cells. Urban dynamics is a field where CA application has been particularly successful (e.g. DEUM—[11]; SLEUTH—[12], and more recently O'Sullivan, 2001; [13]). When the system to be modelled involves heterogeneous entities in more complex situations such as those in social systems, ABM is generally preferred. Agents are defined by their behaviour, can be reactive or cognitive, and interact with other agents and their environment [14]. A review of the use of ABM in ecosystem management can be found in [15]. Software for multi-agent simulations includes CORMAS (Bousquet et al., 1998), NetLogo [16] and GAMA [17].

All three modelling approaches have specific characteristics that can be considered merits or weaknesses depending on the objectives sought. In particular, modellers often need to study a system as a whole, and at the same time decipher how local and intermediate level processes sum up to form the whole system. It is therefore not surprising that there have been attempts to mix or integrate the different approaches. For example, the SME mentioned above can be considered an integration of SD and CA, whereas Clarke [10] explored the origins and key respective contributions of CA and ABM. Schieritz and Milling [18] carried out a detailed comparison of SD and ABM, and reflected on previous promising but still unfulfilled attempts to combine top-down (SD) and bottom-up (ABM) approaches. Since 2008, we have been developing an approach that can be considered intermediate between top-down and bottom-up approaches. The rationale was not to integrate the two types of approaches but rather to focus on their "common denominator" that are the interactions. Any system is described in terms of entities distributed in space that are in interaction with each other, and simulation models of geographical changes are built using **interaction graphs** to explicitly describe processes [19]. The interaction graphs have entities as vertices, and interaction functions attached to their edges. A graph alone can only structure the neighbourhood relationships in a system (which entity is in relation with which other entity) and not the nature of the relations (what happens when entities interact). Nor does it describe how the system evolves with time. Interaction graphs were thus introduced as an extension to the mathematical definition of graphs by allowing (interaction) functions to be applied on the edges simultaneously [20]. These functions are able to access the properties of the entities connected, use and optionally change them, according to the processes being modelled. The interaction graphs are also dynamic in the sense that vertices and edges can be added or removed, and their properties modified during the simulation.

When modelling a system and its dynamics using interaction graphs, one has to imagine what interactions are at play in the system, how they are distributed (spatially, functionally, socially...) and how they can influence the temporal evolution of that system. Such a definition of an interaction graph is very generic. One same concept is used to describe hierarchical relationships (allowing aggregation and disaggregation operations), spatial relationships (from regular grid based neighbourhood, to any other structure issued from vector based geographic information layers or from a continuous spatial reference system), social relationships (by writing socially meaningful semantics in the interaction functions), or more generally any kind of functional based relationship. We combine this genericity with well-chosen operators in the form of a Domain Specific language (see [21], for a review of DSL in ecological modelling) to offer a rich capacity of expression for modelling a wide range of spatially explicit systems and their dynamics. The generally spatial entities used in Ocelet models are represented with data types that are commonly used in Geographical Information Systems (GIS): points, lines, polygons, multi-points, multi-lines and multi-polygons. Interactions between entities result in changes in the state and (spatial) configuration of the entities. The key difference is that, once imported from a GIS data file (e.g. shapefile) into the model, the entities no longer belong to a "GIS layer", and can be interconnected individually through several interaction graphs.

Spatial dynamics models are built within a software environment called the "Ocelet Modelling Platform" (OMP). After a few years of practice and the transition phase between the initial prototype and the current stabilised version of Ocelet (version 1.1), we hereby (Section 2) present the main concepts and features of the software. Three test cases are then briefly described to illustrate the generic disposition of Ocelet (Section 3). Finally, we discuss how the approach can contribute to address scientific questions and also what are the main types of modelling situations that can be tackled (Section 4).

2. Software description

The OMP software environment is built around the Ocelet DSL in order to facilitate model creation and maintenance, code

edition, compilation, simulation run, display and exportation of simulation results. The Ocelet language is used to describe the geographical setup and the processes relevant for a given study. It incorporates specific language elements required for implementing dynamic interaction graphs that are at the heart of the Ocelet modelling approach. The main concepts of the language are given in Section 2.1. Section 2.2 describes how interaction graphs are built and used, and Section 2.3 gives a brief presentation of the Ocelet Modelling Platform.

2.1. Main concepts of the Ocelet language

Entity, *relation* and *scenario* are the main concepts adjoined to the basic features available in most programming languages (e.g. blocks, operations on data types, control loops, conditional branching) to form Ocelet.

An ***entity*** in Ocelet is generally spatial, although this is not necessarily the case. It is defined with a number of properties whose values may change during simulations. A spatial entity has one of its properties to define its spatial extent and position within a spatial reference system. Other non spatial properties of the entities can be of common data types such as String, Integer, Double, and Boolean. Ocelet proposes several data types (geometric types, colour types, date type, user-defined composite types) and collections (group, list, keymap) to facilitate the manipulation and display of spatial and non-spatial information. Entity specific functions can also be defined to access and modify the properties of the entity to express internal processes (those that take place without interactions with other entities).

A ***relation*** is used to describe the semantics that will be attached to the edges of the interaction graphs. After declaring which two types of entities are concerned in the relation, it is possible to define any form of interaction functions that will be applied to pairs of such entity types within the scope of the model. Because the pair of entities is strongly typed, an interaction function can describe precisely which properties are used from the interacting entities and which are affected as a result of the interaction process. Every *relation* thus defined can later be instantiated to hold an interaction graph. Fig. 1 gives a schematic representation of different types of relations that can be modelled in a typical rural landscape. It can be noted that one entity can be involved in several relations.

The ***scenario*** is that part of the model where the initial state of the system is set and the simulation steps scheduled. It contains an ordered sequence of operations that are executed during a simulation run to represent the changes occurring in a geographical area over a period of time. During these operations, entities and relations are instantiated, and functions on the resulting interaction graphs are activated in a specified order to make the system evolve with time. To streamline model writing and maintenance, several scenarios can be defined in the same model. One of them will be considered the main scenario (which has the same name as the project). The other scenarios can be called from the main scenario. No specific mechanism has yet been included to track the properties of the entities or other state variables. Therefore, the scenarios must also contain instructions to save the required variables into a file for display and analysis.

2.2. Building and using explicit dynamic interaction graphs

The downside of the concept of interaction graph being low-level and generic is that building such graphs could be tedious without the help of appropriate tools. Ocelet comes with data format specific interfaces (here called ***datafacer***) that can be used to feed the model with ready for use entities. From tabular-like data formats (such as csv, shapefile, PostGIS) one can obtain an entity for each line or record. A match mechanism is implemented for assigning entity property values directly from attribute or column data. Output datafacers also exist for the three above-mentioned formats, plus another frequently used one, specifically for displaying animated maps in Google Earth (kml format). To help build the edges of the graphs, Ocelet also comes with appropriate functions and operators that use entities types and their property values (including spatial properties). A graph structure can thus be easily built for any kind of spatial or non spatial relationship.

The activation of a function on an interaction graph within a time step will execute that function in a way to mimic simultaneity over the whole interaction graph. That is, each function reads the previous state and writes the next state, such that the result is independent of the order in which the edges of the graph are run through. Inherent to that way of functioning is the possibility for an entity to receive conflicting instructions (e.g. different values assigned to the same property) through the multiple edges that lead to it. As a solution to that problem, we introduced the possibility of defining aggregation functions that would take in all candidate values and deliver one value to be assigned to the property. Depending on the property type concerned, common aggregation functions are available in-built, such as mean, maximum, minimum, for numerical properties (integer or double), but user-defined aggregation functions are also possible. Another interesting modelling mechanism is the possibility of filtering the graph before applying interaction functions. The filter can be defined to retain only the edges that fulfil certain conditions, like for example, those connecting entity pairs having the required combination of property values.

2.3. The Ocelet modelling platform

Ocelet is a compiled DSL. The compilation consists in automatically translating a model written in Ocelet into Java classes using a code generator. The execution of the compiled model then uses the Ocelet java runtime which contains a set of java classes specifically developed for Ocelet, as well as several other functions from known open-source libraries such as the Java Topology Suite (JTS) and Geotools. In the OMP environment, these software dependencies are rendered transparent to the user. Fig. 2 shows the main features of the OMP interface, built with Eclipse RCP and composed of four frames. In the middle one there is a text editor with syntax highlighting. Code errors are underlined in red. The left frame contains the project navigator. A project contains

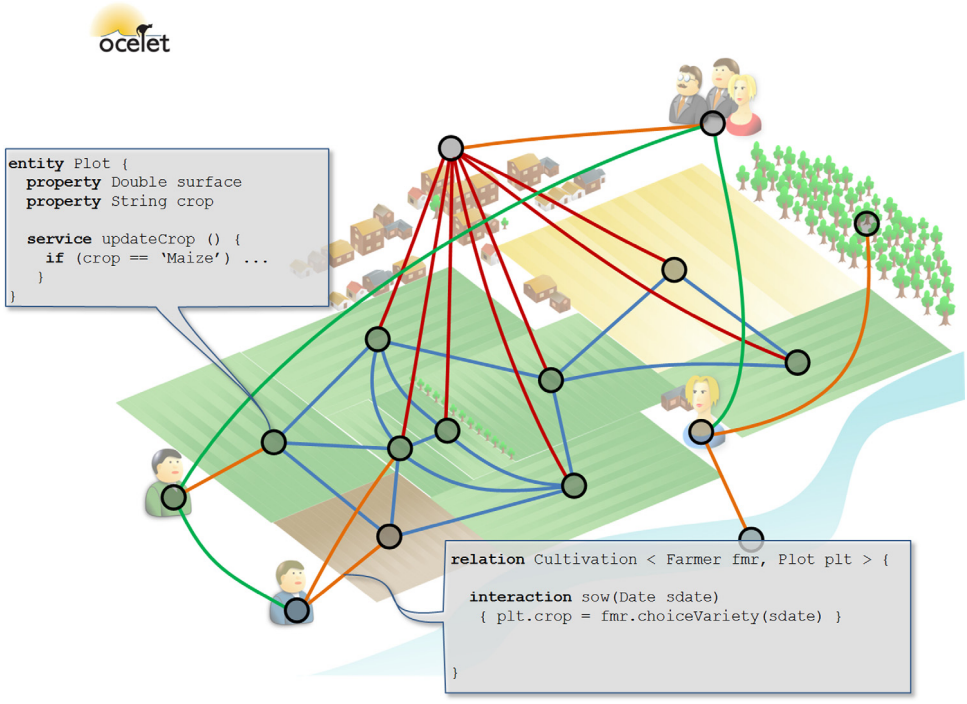

Fig. 1. Schematic diagram showing landscape elements in interaction. Circles are the vertices, and coloured lines, the edges, of a graph. Circles represent agricultural plot, river, forest, town, farmer, environmental manager and decision maker entities. Lines represent spatial (blue), functional (orange), hierarchical (red) and social (green) relationships between the entities, on which interaction functions can be applied. Examples of a Plot entity definition and a Cultivation relation definition are also given. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

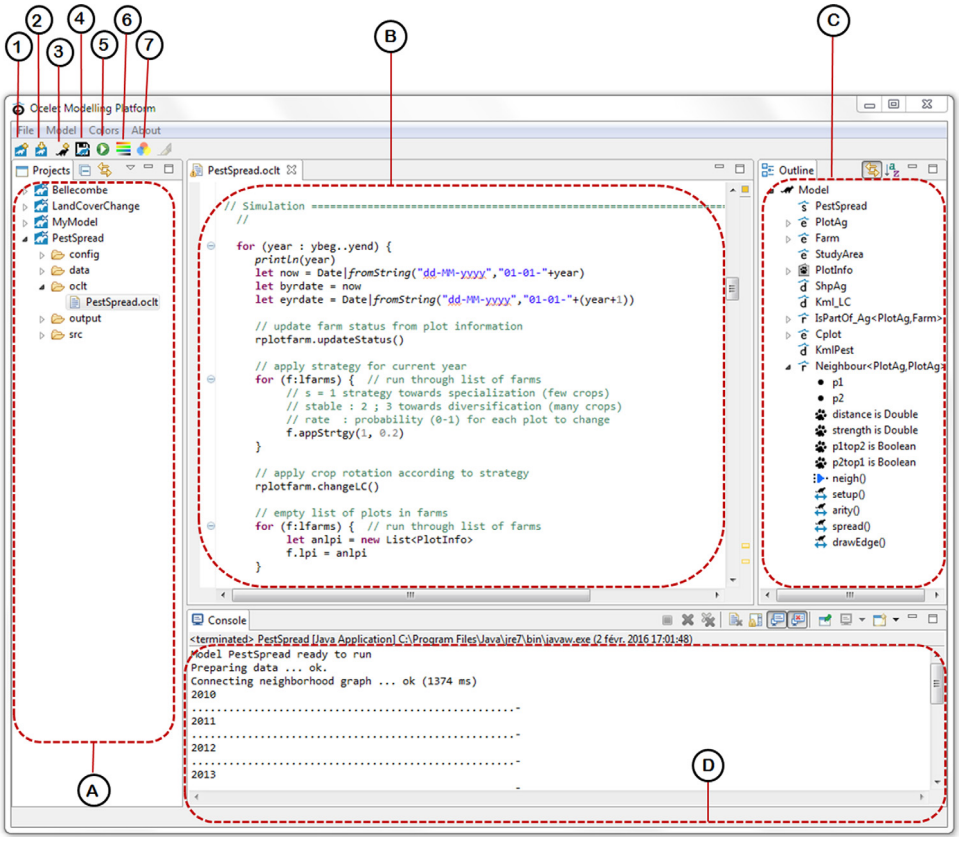

Fig. 2. Ocelet Modelling Platform interface with windows A–D as follows: A—project explorer, B—model text editor, C—model outline and D—console window. Buttons 1–6 are as follows: 1—create a new Ocelet project, 2—import an existing Ocelet project, 3—create a new document, 4—save a document, 5—run a simulation, 6—visualise colour palettes, and 7—choose a colour.

the description of a model, with all the input data and configuration files organised in sub-folders. The right frame gives the outline of the model that is displayed in the middle frame. It contains the model structure (the entities, relations, scenarios and datafacers) and allows a quick navigation within the model code. At the bottom we have the console where messages produced during the simulation are displayed. Several buttons (Fig. 2: 1–7) are provided for the most used functions. These are also found in the drop down menu at the top.

3. Illustrative examples

Three examples are briefly presented to illustrate different types of possible applications: (i) colonisation of an open space by a plant species, (ii) land cover change in an agricultural landscape and (iii) pest invasion in the same landscape. These examples are quite simple but they contain model elements that can be reused in very diverse situations. The models are briefly explained below. Screen captures of output kml files as displayed in Google Earth at different dates are also shown. Links to model codes, input and output files are provided in the "Examples" tab of the website given in Table 2.

- *Colonisation by an introduced plant species*
  A new plant species is introduced in an empty space and therefore faces no competition apart from itself. Simple rules are used for simulating plant growth through different phenological phases. In its adult phase, a plant can produce seeds that are spread in its close environment. Some of the seeds germinate, grow and reach the adult phase before producing their own seeds. A local constraint hinders germination and growth when plant density is too high. (See Fig. 3a.)
- *Land cover change in an agricultural landscape*
  Models are frequently used by researchers and professionals to study the drivers of land cover and land use change (e.g. review by [4]). In the present example, we model agricultural plots and the farms to which they belong. Farmers manage crop rotations and fallow in their plots according to individual "strategies" towards fewer or more crop species. The time step is the year. Model parameters can be changed to simulate different scenarios of farm diversification or specialisation, following concerted or independent management choices. (See Fig. 3b.)
- *Pest spreading in an agricultural landscape*
  In the previous land cover change model, we have added the possibility of some plots being attacked by insect pests that can spread to neighbouring plots through proximity. Some land cover types can be used by the insects as favourable environments for reproduction or food, whereas others are unfavourable environments to be avoided. The time step is a week. The model is initialised with pests present in some plots with a land cover favourable for pest reproduction. Neighbouring plots containing food for the pests are affected and the pests are gradually dispersed into the landscape. Different land cover change scenarios can thus be tested for their capacity to resist pest spreading. (See Fig. 3c.)

4. Discussion

When writing an Ocelet model, model specification is carried out at three levels: (i) the individual level when building entities with their properties and services, (ii) the interaction level, when designing relations with their interaction graphs, filter, aggregation and interaction functions, and (iii) the system level and its dynamics, with the scenario describing the initial state of the system and how it evolves with time. The nature of the interactions needed in a model can be quite diverse. Representing how hierarchical, functional, spatial or even social relations interfere is often necessary when trying to understand how a system behaves. In Ocelet, interaction graphs are used to treat these diverse interactions in a conceptually and computationally unified way. It ensues that processes expressed from different points of view and spatial levels can be integrated in the same model. For example, functional relations can link farmers to their agricultural plots to express a farming point of view, whereas, spatial relations between agricultural plots would be relevant in a pest management point of view. Likewise, entities can be defined at different spatial levels, with agricultural plots grouped into farms, and farms associated in agricultural cooperatives. Farmers can be made to share innovations through geographical and social proximity, and at the same time compete against each other for resources.

The choice of interaction graphs suggest that some types of models may not be conveniently built using Ocelet. For example, Ocelet entities may have similarities with reactive agents, but they do not possess the same level of autonomy as more elaborate agents. Therefore agent based models involving autonomous cognitive agents will be more difficult to build than those with only reactive agents. Also, since interaction functions are applied synchronously and simultaneously to all the selected edges of interaction graphs, models where the dynamics can be described using synchronous time steps are preferred. It may not be convenient to use Ocelet for building asynchronous event based models.

5. Conclusions and perspectives

One of our main motivations when designing Ocelet was to offer to modellers an as large as possible capacity of expression, with (a) no predefined representation of space and the possibility to mix more than one (grid, vector, continuous), and (b) no predefined form of relationship, where most modelling tools have their own relationship meta-model (e.g. agent-population hierarchy, topological adjacency neighbourhood) which can sometimes be constraining for the modelling exercise. We believe that having only one same generic concept to hold any kind of interaction, with the support of a domain specific language, is a good compromise between capacity of expression and ease of use. It also provides appropriate concepts and associated tools to help modellers integrate points of views from different scientific disciplines into one same model.

Given the generic nature of interaction graphs, Ocelet can be used in a wide range of applications involving multi-disciplinary issues. For example, Ocelet has been successfully

Fig. 3a. Screen captures of output kml files for 'Colonisation by an introduced species' as displayed in Google Earth.

Fig. 3b. Same as 3a but for 'Land cover change in an agricultural landscape'.

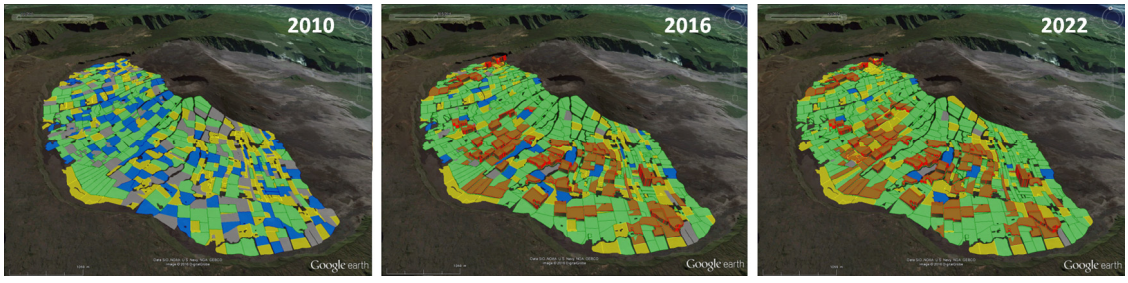

Fig. 3c. Same as 3b but for 'Pest spread' in the same landscape.

used for experimenting land use policy simulation scenarios in a participatory approach [22], studying farming systems evolution from plot to regional scale [23], and modelling mangrove coastline changes [24]. User feedbacks should help to enrich the language and the platform with new features, although the initial concepts can be considered stable. A foreseen addition to the version 1.1 presented here will be an important optimisation work on the capacity to mix large grid, continuous and vector representations of space in the same model. An efficient treatment of a large number of grid cells (typically the number of pixels in a satellite image) is being tested [25] and is planned to be incorporated in the next version of Ocelet.

Acknowledgements

This work was partially funded by the French National Research agency (ANR) through the STAMP (ANR-07-BLAN-0121 http://wiki.ocelet.fr/) and DESCARTES (ANR 11-AGRO-002-01 http://www.projet-descartes.fr/) projects. We also thank two anonymous reviewers whose comments have helped improve the paper.

References

[1] Schubert J, Moradi F, Asadi H, Luotsinen L, Sjöberg E, Hörling P, et al. Simulation-based decision support for evaluating operational plans. Oper Res Perspect 2015;2:36–56.
[2] Schmidt-Lainé C, Pavé A. Environment: modelling and models to understand, to manage and to decide in an interdisciplinary context. Nat Sci Soc 2002;10s1:5–25. http://dx.doi.org/10.1016/S1240-1307(02)80131-5.
[3] Baker W. A review of models of landscape change. Landscape Ecol 1989; 2(2):111–33.
[4] Agarwal C, Green GM, Grove JM, Evans TP, Schweik CM. A review and assessment of land-use change models: dynamics of space, time, and human choice. Gen. Tech. Rep. NE-297. Newtown Square, PA: U.S. Department of Agriculture, Forest Service, Northeastern Research Station; 2002. 61 p.
[5] Laniak GF, Olchin G, Goodall J, Voinov A, Hill M, Glynn P, et al. Integrated environmental modeling: a vision and roadmap for the future. Environ Modelling Softw 2013;39:3–23.
[6] Forrester JW. Principles of systems. Mass: Wright-Allen Press Cambridge; 1968.
[7] Forrester JW. Urban dynamics. Cambridge, Mass: M.I.T. Press; 1969.
[8] Costanza R. Simulation modeling on the macintosh using stella. BioScience 1987;37:129–32.
[9] Eberlein R, Peterson D. Understanding models with Vensim TM. European J Oper Res 1992;59:216–9.

[10] Clarke KC. Cellular automata and agent-based models. In: Fischer MM, Nijkamp P, editors. Handbook of Regional Science. Berlin, Heidelberg: Springer-Verlag; 2014 [chapter 62].
[11] Batty M, Xie Y. From cells to cities. Environ Plan B 1994;21:31–48.
[12] Clarke K, Hoppen S, Gaydos L. A self-modifying cellular automaton model of historical urbanization in the San Francisco Bay area. Environ Plan B 1997;24:247–61.
[13] Barreira-González P, Gómez-Delgado M, Aguilera-Benavente F. From raster to vector cellular automata models: A new approach to simulate urban growth with the help of graph theory. Comput Environ Urban Syst 2015;54:119–31.
[14] Ferber J. Multi-agent systems, an introduction to distributed artificial intelligence. Reading: Addison-Wesley; 1999.
[15] Bousquet F, Le Page C. Multi-agent simulations and ecosystem management: a review. Ecol Model 2004;176(3–4):313–32.
[16] Wilensky U. NetLogo. Center for Connected Learning and Computer-Based Modeling. Evanston, IL: Northwestern University; 1999. http://ccl.northwestern.edu/netlogo/.
[17] Grignard A, Taillandier P, Gaudou B, Vo D-A, Huynh N-Q, Drogoul A. GAMA 1.6: Advancing the art of complex agent-based modeling and simulation. In: PRIMA 2013: Principles and practice of multi-agent systems. Lecture Notes in Computer Science, vol. 8291. Springer; 2013. p. 117–31.
[18] Schieritz N, Milling PM. Modeling the forrest or modeling the trees: A comparison of system dynamics and agent based simulation. In: Proceedings of the XXI International conference of the system dynamics society; 2003.
[19] Degenne P, Lo Seen D, Parigot D, Forax R, Tran A, Ait Lahcen A, et al. Design of a domain specific language for modelling processes in landscapes. Ecol Model 2009;220(24):3527–35.
[20] Degenne P, Ait Lahcen A, Curé O, Forax R, Parigot D, Lo Seen D. Modelling with behavioural graphs. Do you speak Ocelet? In: International congress on environmental modelling and software, July 5–8, Ottawa, Ontario, Canada; 2010. http://www.iemss.org/iemss2010/.
[21] Holst N, Belete GF. Domain-specific languages for ecological modelling. Ecol Inform 2015;27:26–38.
[22] Lestrelin G, Augusseau X, David D, Lagabrielle E, Lo Seen D, Bourgoin J. et al. Collaborative landscape research in Reunion Island: Participatory modelling and spatial simulation to foster dialogue and knowledge integration. In: International association for landscape ecology (IALE) 2013 European congress: Changing European landscapes, Manchester (UK), 09–12 September; 2013.
[23] Jahel C, Baron C, Vall E, Karambini M, Castets M, Coulibaly K, Bégué A, Lo Seen D. Spatial modelling of agro-ecosystem dynamics across scales: A case in the cotton region of West-Burkina Faso. Agricultural Systems 2016 (in press).
[24] Proisy C, Degenne P, Anthony EJ, Berger U, Blanchard E, Fromard F, et al. A multiscale simulation approach for linking mangrove dynamics to coastal processes using remote sensing observations. In: Vila-Concejo A, Bruce E, Kennedy DM, McCarroll RJ, editors. Proceedings of the 14th International coastal symposium (Sydney, Australia). Journal of Coastal Research, Special Issue, No. 75. Coconut Creek (Florida); 2016. p. 810–4. ISSN 0749-0208.
[25] Castets M, Degenne P, Poncelet P, Lo Seen D. Integrating raster and vector spatial representations with interaction graphs for multi-scale environmental simulations. In: International congress on environmental modelling and software, June 15–19, San Diego, California, USA; 2014. http://www.iemss.org/iemss2014/.
[26] Costanza R, Voinov A. Landscape simulation modeling. A spatially explicit, dynamic approach. New York: Springer-Verlag; 2004.